

*Lucrare pentru susținerea*

*examenului de atestat profesional*

**FriendlyFire**

Profesori îndrumători:

* Belbe Maria

Elev: Șpan Claudiu-Beniamin

Anul școlar 2019-2020

1

Cuprins:

1. Introducere.....................................................3
2. Prezentarea aplicației.................................4
3. Utilizarea aplicației.......................................5
4. Codul sursă.....................................................7
5. Bibliografie.......................................................11

2

1.Introducere:

Un joc video este un joc electronic în care se interacționează cu o interfață grafică pentru a genera răspuns vizual pe un ecran. Jocurile video în general au un sistem de recompensare a utilizatorului, de obicei se ține scorul, acest sistem depinzând de îndeplinirea unor anumite obiective în joc.

Termenul [**joc pe calculator**](https://ro.wikipedia.org/wiki/Joc_pe_calculator) sau "[joc pe PC](https://ro.wikipedia.org/wiki/Joc_pe_PC)" se referă la un joc care este jucat pe un [PC](https://ro.wikipedia.org/wiki/Computer_personal), unde [monitorul](https://ro.wikipedia.org/wiki/Monitor) este principalul mijloc de feedback și care folosește ca dispozitiv de control un [periferic](https://ro.wikipedia.org/wiki/Periferic) de intrare, de obicei butonarea unui [joystick](https://ro.wikipedia.org/wiki/Joystick) (jocuri din arcade-uri), o combinație [tastatură](https://ro.wikipedia.org/wiki/Tastatur%C4%83) & [maus](https://ro.wikipedia.org/wiki/Maus)/[trackball](https://ro.wikipedia.org/wiki/Trackball) sau un [controller](https://ro.wikipedia.org/wiki/Controller), ori o combinație dintre cele de mai sus.

Un **platformer** (sau **joc de platformă**) este un [joc video](https://ro.wikipedia.org/wiki/Joc_video) în care personajul controlat de jucător sare pe platforme suspendate sau peste obstacole pentru a avansa în joc.

La un moment dat, jocul de platformă era cel mai popular gen de joc video. În perioada de vârf a popularității lor, se estimează că între un sfert și o treime din jocurile de consolă erau de acest gen. Nici un alt gen nu a reușit să atingă o pondere la fel de mare. Din 2006, genul a devenit mai puțin popular, cu doar 2% din jocuri fiind de acest gen, în comparație cu 15% în anul 1998,[[1]](https://ro.wikipedia.org/wiki/Platformer#cite_note-detailed_cross-examination-1)

Cele mai cunoscute platformere sunt: seria de jocuri [Mario](https://ro.wikipedia.org/wiki/Mario_(serie_de_jocuri)), [Sonic](https://ro.wikipedia.org/wiki/Sonic_the_Hedgehog_(joc_video_din_2006)), [Metroid](https://ro.wikipedia.org/wiki/Metroid), [Contra](https://ro.wikipedia.org/w/index.php?title=Contra&action=edit&redlink=1), [Jazz Jackrabbit 2](https://ro.wikipedia.org/wiki/Jazz_Jackrabbit_2).

3

2.Prezentarea aplicației:

Am realizat o aplicație folosind motorul grafic numit Unity, împreună cu programarea orientată pe obiecte in limbajul C#, rezultând într-un joc 1v1 de tipul platformer 2d shooter, având posibilitatea de a alege 2 scenarii: o hartă de iarnă, și una în junglă

La baza programului sunt definite următoarele obiecte care interacționează între ele:

* Primul jucator (Player1)
* Al doilea jucator(Player2)
* Bulgărele de zapada/Săgeata
* Efectul de distrugere
* Decorul
* Platformele
* Sunetul de fundal
* Sunetul de aruncare
* Sunetul de lovit
* Bara cu numărul de vieți ale fiecarui jucător

De asemenea sunt integrate diferite ”camere de joc” sau scene, locul unde obiectele amintite mai sus interacționează:

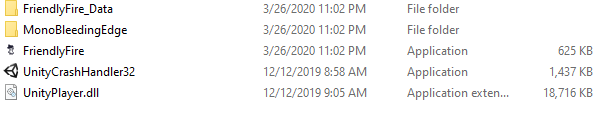
* Meniul Principal
* Nivelul cu zăpadă
* Nivelul în junglă
* Ecranul din momentul pierderii tuturor vieților unui jucător

4

3.Utilizarea aplicației:

Pentru rularea jocului se procedează astfel:

1. Se acceaseaza folderul ”Atestat”
2. În folderul ”Atestat” se selectează executabilul ”Friendly Fire”



Meniul principal:

Prin executarea dublu click asupra aplicației va apărea urmatorul ecran:

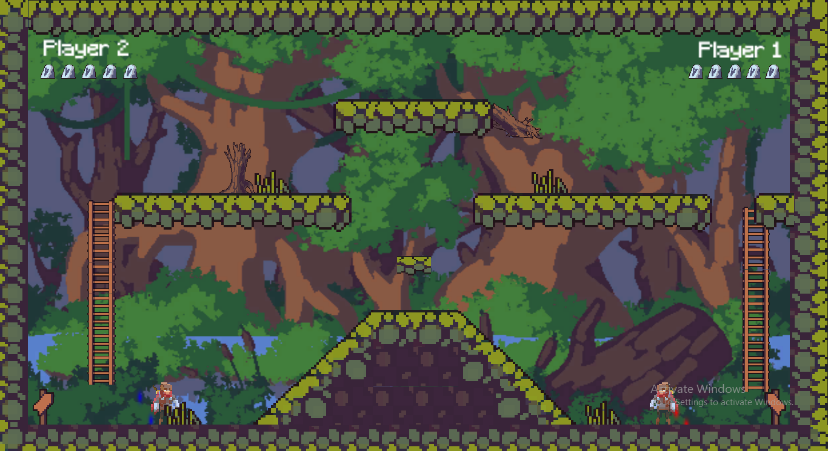
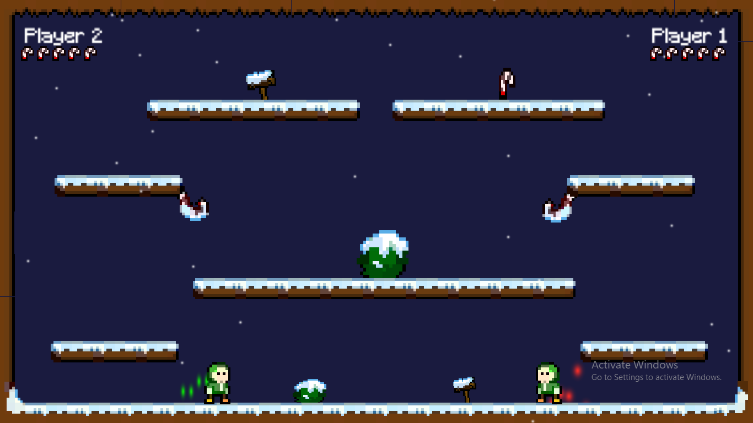


Se observă meniul principal al jocului unde sunt prezentate următoarele:

titlul jocului, 2 butoane pentru hărți diferite de joc, butonul de ieșire, iar in colțurile de jos sunt afișate instrucțiuni pentru controalele fiecărui jucător.

5

Ecranul de joc(cele 2 nivele):



Prin selectarea unui nivel de joc, ne redirecționează către una dintre cele 2 hărți de joc, în funcție de preferințe. Aici observăm în partea

dreaptă primul jucător(Player1) având particule roșii împreună cu numărul de vieți , respectiv al doilea jucător (Player2).

Caracterele se controlează folosind instrucțiunile prezentate în meniu(vezi ”Meniul Principal”).

Scopul jocului este de a putea învinge oponentul prin eliminarea tuturor vieților acestuia, prin explorarea hărții și prin țintirea lui cu bulgărele/săgeata care îl va costa pe inamic un punct de viață.

După ce un jucător își pierde toate viețile distribuite în colțul de sus, se va afișa un ecran cu numele câștigătorului împreună cu alte instrucțiuni.



6

4.Codul sursă:

Se va afișa codul sursă pe obiecte pentru o mai bună

înțelegere a algoritmilor:

* Controlul caracterului(Player1/Player2)
* public float moveSpeed;
* public float jumpForce;
* public KeyCode left;
* public KeyCode right;
* public KeyCode jump;
* public KeyCode throwBall;
* private Rigidbody2D theRB;
* public Transform groundCheckPoint;
* public float groundCheckRadius;
* public LayerMask whatIsGround;
* public bool isGrounded;
* private Animator anim;
* public GameObject snowBall;
* public Transform throwPoint;
* public AudioSource throwSound;
* public float xscale;
* public float yscale;
* public float zscale;

Am setat fiecare variabilă care aparține jucătorului respectiv:

viteza de mișcare, forța săriturii, codurile tastaturii, corpul jucătorului, raza în care se detectează coliziunea cu pamîntul, animațiile, proiectilul, marimea jucătorului.

* void Start()

{

xscale = 0.6f;

yscale = 0.6f;

zscale = 0.6f;

theRB = GetComponent<Rigidbody2D>();

anim = GetComponent<Animator>();

}

Am inițializat cîteva variabile.

void Update()

isGrounded = Physics2D.OverlapCircle(groundCheckPoint.position, groundCheckRadius, whatIsGround);

if(Input.GetKey(left))

{

theRB.velocity = new Vector2(-moveSpeed, theRB.velocity.y);

}

else

if(Input.GetKey(right)

7

theRB.velocity = new Vector2(moveSpeed, theRB.velocity.y);

}

else

{ theRB.velocity = new Vector2(0, theRB.velocity.y);

}

if(Input.GetKeyDown(jump) && isGrounded)

{

theRB.velocity = new Vector2(theRB.velocity.x, jumpForce);

}

if(Input.GetKeyDown(throwBall))

{

GameObject ballClone = (GameObject) Instantiate(snowBall, throwPoint.position, throwPoint.rotation);

ballClone.transform.localScale = transform.localScale;

anim.SetTrigger("Throw");

throwSound.Play();

}

if (theRB.velocity.x < 0)

{

transform.localScale = new Vector3(-xscale, yscale, zscale);

}

else if(theRB.velocity.x>0)

{

transform.localScale = new Vector3(xscale, yscale, zscale);

}

anim.SetFloat("speed",Mathf.Abs( theRB.velocity.x));

anim.SetBool("Grounded", isGrounded);

Codul efectiv al controlului caracterului: schimbarea direcției de mers + aruncarea proiectilului.

* Managerul de joc(evenimentele ce se întîmplă pe parcursul jocului)
* public class GameManager : MonoBehaviour

{

* public GameObject player1;
* public GameObject player2;
* public int P2Life;
* public int P1Life;
* public GameObject[] p1Life;
* public GameObject[] p2Life;
* public GameObject P1Win;
* public GameObject P2Win;
* public AudioSource hurt;
* public string mainMenu;

Am inițializat cîteva variabile.

* void Update(){

8

* + if (P1Life <= 0)

{

* + - player1.SetActive(false);
    - P2Win.SetActive(true);
    - Time.timeScale = .3f;}
  + if (P2Life <= 0)

{

* + - player2.SetActive(false);
    - P1Win.SetActive(true);
    - }

Codul pentru afișarea cîștigătorului.

* + if(Input.GetKeyDown(KeyCode.R))

{

* + - Time.timeScale = 1f;
    - SceneManager.LoadScene(SceneManager.GetActiveScene().name);

}

* + if(Input.GetKeyDown(KeyCode.Escape))

{

* + - Time.timeScale = 1f;
    - SceneManager.LoadScene(mainMenu);

}

Codul pentru reinițializarea nivelului respective întoarcerea la meniul principal.

* public void hurtP1()

{

* + P1Life -= 1;
  + for (int i = 0; i < p1Life.Length; i++)

{

* + - if (P1Life > i)

{

* + - p1Life[i].SetActive(true);
    - }
    - else
    - {
    - p1Life[i].SetActive(false);}

}

* + hurt.Play();}
* public void hurtP2()

{

* + P2Life -= 1;
  + for (int i = 0; i < p2Life.Length; i++)
  + {
    - if (P2Life > i)
    - {
    - p2Life[i].SetActive(true);
    - }
    - else
    - {
    - p2Life[i].SetActive(false);

}

}

* + hurt.Play();}

Codul pentru calcularea și afișarea vieților rămase.

9

* Proiectilul
* public class Snowball : MonoBehaviour

{

* public float ballSpeed;
* private Rigidbody2D theRB;
* public GameObject SnowballEffect;
* // Start is called before the first frame update
* void Start()

{

* theRB = GetComponent<Rigidbody2D>();

}

Inițializarea variabilelor

* void Update()

{

* theRB.velocity= new Vector3(ballSpeed \* transform.localScale.x, 0);

}

* void OnTriggerEnter2D(Collider2D other)

{

* if( other.tag == "Player1")

{

* FindObjectOfType<GameManager>().hurtP1();

}

* if (other.tag == "Player2")

{

* FindObjectOfType<GameManager>().hurtP2();

}

* Instantiate(SnowballEffect, transform.position, transform.rotation);
* Destroy(gameObject);

}

}

Fixarea vitezei proiectilului + detectarea coliziunii cu jucatorul sau decorul.

* Meniul principal:
* public class Menu : MonoBehaviour

{

* public string snowLevel;

public string jungleLevel;

* public void StartSnowLevel()

{

* SceneManager.LoadScene(snowLevel);

}

* public void StartJungleLevel()

{

* SceneManager.LoadScene(jungleLevel);}

10

* public void QuitGame()

{

* Application.Quit();

}

}

Acțiunile meniului principal(selectarea nivelelor,ieșirea din joc)

* Alte surse programabile:

Am folosit editorul ”Unity” pentru a configura harta împreuna cu decorul, animațiile jucătorului și alte resurse oferite de acesta.

5.Bibliografie:

* + - * <https://ro.wikipedia.org/wiki/Platformer>
      * <https://ro.wikipedia.org/wiki/Joc_video>
      * [https://itch.io/game-assets](https://itch.io/game-assets(siteu-l)

-Proiect realizat utilizând programul Unity-

11